EXAMINER'S AMENDMENT

1. An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

2. Authorization for this examiner's amendment was given in a telephone interview with Ms. Tammy M. Pennington (Reg. No. 61,223) on 2/25/2009.

3. Amend the claims as the following paragraph:

1. (currently amended): A method for establishing a device driver in an open source operating system, comprising the steps of:

providing a device driver having ~~at least one~~ multiple pre-compiled ~~module~~ modules in executable form and a service layer in open source form, wherein each of the multiple pre-compiled modules is associated with a hardware architecture of the open source operating system; ~~and~~

compiling the service layer against the kernel of the open source operating system after each modification to the kernel of the open source operating system, wherein the step of compiling the service layer against the kernel comprises the step of associating the naming convention of function calls in the kernel to the naming convention of expected function calls in the device driver;

wherein the compiled service layer acts as an interface between the kernel of the operating system and the ~~at least one~~ multiple pre-compiled executable ~~module~~ modules of the device driver, such that the kernel cannot access proprietary information of the multiple pre-compiled executable ~~module~~ modules and the compiled service layer is operable to send and receive function calls that are named according to the same naming convention[[.]];

wherein the naming convention comprises the use of a suffix for the naming of function calls, the suffix providing a naming convention that is specific to the kernel of the operating system.

2. (previously canceled.)

3. (currently amended): The method for establishing a device driver in an open source operating system of claim 1, further comprising the step of linking the compiled service layer to the multiple pre-compiled executable modules ~~module~~ in executable form to form the device driver.

4. (original): The method for establishing a device driver in an open source operating system of claim 3, further comprising the step of storing the device driver in memory.

5. Canceled.

6. Canceled.

7. Canceled.

8. (previously amended): A computer system comprising:

a processor;

a memory;

an open source operating system having a kernel;

a device driver, the device driver comprising,

an executable module compiled from an open source service layer, ~~and~~

~~at least one~~ multiple pre-compiled executable ~~module~~ modules, wherein each of the multiple pre-compiled executable modules is associated with a hardware architecture of the computer system,

wherein the executable module compiled from the open source service layer provides an interface between the kernel of the operating system and the ~~at least one~~ multiple pre-compiled executable ~~module~~ modules, such that the kernel cannot access proprietary information of the multiple pre-compiled executable ~~module~~ modules, such that the executable module compiled from the open source service layer receives kernel-specific function calls from the kernel of the operating system, wherein the executable module is compiled from the open source service layer following each modification to the kernel of the operating system, and wherein the kernel of the operating system and the executable module compiled from the open source service layer send and receive function calls according to the same naming convention[[.]], and

wherein the naming convention comprises the use of a suffix for the naming of function calls, the suffix providing a naming convention that is specific to the kernel of the operating system.

9. (original): The computer system of claim 8, wherein the device driver is loaded in memory of the computer system.

10. Canceled.

11. (previously canceled).

12. Canceled.

13. (currently amended): A method for loading a device driver in a computer system having an open source operating system, comprising the steps of:

compiling an open source service layer against the kernel of the operating system following a modification to the kernel of the operating system; ~~and~~

linking the compiled service layer to a set of precompiled driver modules, each of the precompiled driver modules being associated with a hardware architecture of a computer system;

wherein the compiled service layer provides an interface between the kernel of the operating system and the precompiled driver modules, such that the kernel cannot access proprietary information of the pre-compiled executable module, and wherein the kernel of the operating system and the compiled service

layer is operable to send and receive function calls that are named according to the same naming convention[[.]];

recompiling the service layer if it is determined that the kernel of the operating system has been modified, wherein the recompiled service layer is operable to send and receive function calls that are named according to the same naming convention; and

relinking the recompiled service layer to a set of precompiled driver modules.

14. (previously canceled).

15. (original): The method for loading a device driver of claim 13, further comprising the step of recompiling the open source service layer if it is determined that the kernel of the open source service layer has been modified.

16. (previously amended): The method for loading a device driver of claim 13, further comprising the step of linking the recompiled service layer to the set of precompiled driver modules.

17. (original): The method for loading a device driver of claim 13, further comprising the step of determining, prior to compilation of the open source service layer, whether a precompiled device driver exists that is associated with the kernel of the operating system and loading the precompiled device driver if such a device driver exists.

18. (original): The method for loading a device driver of claim 13,

wherein the function calls passed between the kernel of the operating system and the compiled open source service layer are not specific to the hardware architecture of the computer system; and

wherein the function calls passed between the compiled open source service layer and the precompiled driver modules are specific to the hardware architecture of the computer system.

19. Canceled.

20. Canceled.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to PHUONG N. HOANG whose telephone number is (571)272-3763. The examiner can normally be reached on Monday - Friday 9:00 am to 5:30 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng A. An can be reached on 571-272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Meng-Ai An/
Supervisory Patent Examiner, Art Unit 2195

/P. N. H./
Examiner, Art Unit 2194